Санкт-Петербургский Государственный Университет Аэрокосмического приборостроения

ФАКУЛЬТЕТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| преподаватель |  |  |  | И.Г. Бартасевич |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНЫХ РАБОТАХ |
|  |
| по дисциплине: Основы алгоритмизации и программирования |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТГР. № | С021к |  |  |  | Панков В.Д |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021/2022

**Лабораторная работа №1**

**Линейные алгоритмы**

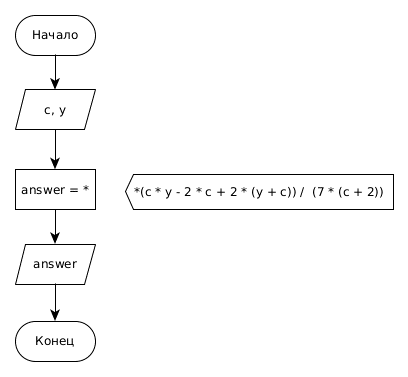
**Цель работы:** научиться строить алгоритмы и писать программы линейной структуры.

**Вариант 22**

**Текстовое описание решение задачи**:

Принять два числа c и y и вычислить по формуле результат.

**Блок-схема**:



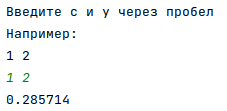
**Программа на языке С++**:

#include <iostream>  
#include <math.h>  
  
using namespace std;  
  
int main() {  
 setlocale(**LC\_ALL**, "");  
 double c, y;  
 cout << "Введите с и y через пробел\nНапример:\n1 2" << endl;  
 cin >> c >> y;  
 double answer = (c \* y - 2 \* c + 2 \* (y + c)) / (7 \* (c + 2));  
 cout << answer << endl;

return 0;

}

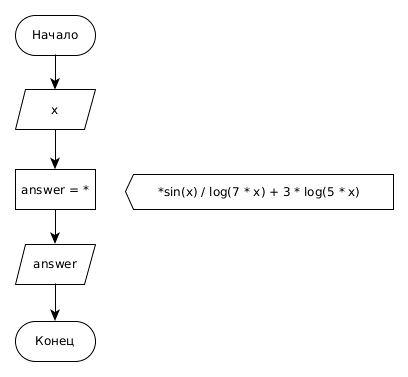
**Демонстрация результата работы программы**:



**Текстовое описание решение задачи**:

Принять число x и вычислить по формуле результат.

**Блок-схема:**



**Программа на языке С++**:

#include <iostream>  
#include <math.h>  
  
using namespace std;  
  
int main() {  
 setlocale(**LC\_ALL**, "");  
  
 double x;  
 cin >> x;  
 double result = sin(x) / log(7 \* x) + 3 \* log(5 \* x);

cout << result << endl;

return 0;  
}

**Демонстрация результата работы программы**:

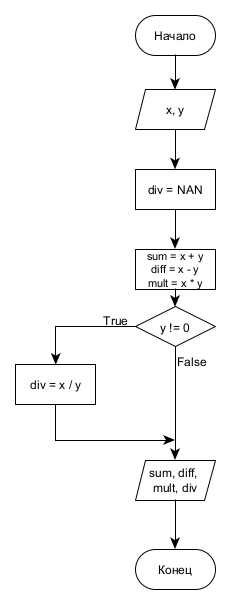


3) Даны два действительных числа x и y. Вычислить их сумму, разность, произведение и частное.

**Текстовое описание решение задачи**:

Принять два числа x и y, и вычислить их сумму, разность, произведение и частное. Также перед вычислением частного, мы должны проверить не равняется ли y 0.

**Блок-схема:**



**Программа на языке С++**:

#include <iostream>  
#include <math.h>  
  
using namespace std;  
  
int main() {  
 setlocale(**LC\_ALL**, "");  
  
 double x, y;  
 cin >> x >> y;  
 double sum, diff, mult, div = **NAN**;  
 sum = x + y;  
 diff = x - y;  
 mult = x \* y;  
 if (y != 0){  
 div = x / y;  
 }  
 cout << sum << " " << diff << " " << mult << " " << div << endl;  
 return 0;  
}

**Демонстрация результата работы программы**:



**Вывод:**

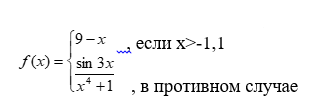
Я научился строить алгоритмы и писать программы линейной структуры.

**Лабораторная работа №2**

**Алгоритмы с ветвлением**

**Цель работы:** научиться строить алгоритмы и писать программы с ветвлением.

**Вариант 22**

1. 

**Текстовое описание решение задачи**:

Принять одно число(x) и проверить является ли оно больше -1.1. Если да, то выполнить 9 – x, иначе sin(x) / (x \* x - 9).

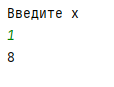
**Блок-схема**:

****

**Программа на языке С++**:

#include **<iostream>**#include **<cmath>**#include **<windows.h>  
  
using namespace** std;  
  
**int** main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
 **float** x, res;  
 cout << **"Введите x"** << endl;  
 cin >> x;  
 res = x > -1.1 ? 9 - x : sin(x) / (x \* x - 9);  
 cout << res << endl;  
 **return** 0;  
}

**Демонстрация результата работы программы**:



**2) Грузовой автомобиль выехал из одного города в другой со скоростью v1 км/ч. Через t часов в этом же направлении выехал легковой автомобиль со скоростью v2 км/ч. Составить программу, определяющую, догонит ли легковой автомобиль грузовой через t1 ч после своего выезда.**

**Текстовое описание решение задачи**:

Ввести значения v1, v2, t и t1. Дальше вычислить расстояние которое проехал грузовой автомобиль по формуле v1 \* (t1 + t) и легковой автомобиль по формуле v2 \* t1.

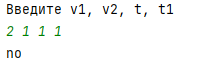
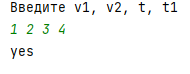
**Блок-схема**:



**Программа на языке С++**:

#include **<iostream>**#include **<windows.h>  
  
using namespace** std;  
  
**int** main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
 **float** v1, v2, t, t1;  
 cout << **"Введите v1, v2, t, t1"** << endl;  
 cin >> v1 >> v2 >> t >> t1;  
 cout << (v1 \* (t1 + t) <= v2 \* t1 ? **"yes"** : **"no"**) << endl;  
 **return** 0;  
}

**Демонстрация результата работы программы**:



**3) Считая, что стандартные функции sin*x* и cos*x* применимы только к аргументам из отрезка [0, π/2], вычислить y=sin*х* для произвольного числа *х*;**

**Текстовое описание решение задачи**:

Принять одно число, проверять входит ли это число в диапазон [0, π/2] и вычислить y = sin x. Иначе вычесть из x 2 \* PI.

**Блок-схема**:



**Программа на языке С++**:

#include "stdafx.h"

#include <iostream>

#include <cmath>

#include <windows.h>

using namespace std;

const double pi\_2 = 1.57079632679489661923;

int main() {

setlocale(LC\_ALL, "");

float x;

cout << "Введите x" << endl;

cin >> x;

if (x >= 0 && x <= pi\_2)

{

cout << sin(x) << endl;

} else

{

cout << sin(x - (pi\_2 \* 4) \* int(x / pi\_2)) << endl;

}

system("pause");

return 0;

}

**Демонстрация результата работы программы**:



**Лабораторная работа №3**

**Оператор множественного выбора – переключатель**

**Вариант 22**

Пусть элементами равностороннего треугольника являются: 1 — сторона а; 2 — площадь s; 3 — высота h; 4 — радиус r вписанной окружности, 5 — радиус описанной окружности R. По заданным номеру и значению соответствующего элемента вычислить значения всех остальных элементов треугольника.

**Текстовое описание решение задачи**:

**Блок-схема**:

****

**Программа на языке С++**:

#include <iostream>

#include <cmath>

#include <stdio.h>

using namespace std;

int main(){

short int number;

cin >> number;

double h = 0;

double a = 0;

double S = 0;

double r = 0;

double R = 0;

switch (number)

{

case 1:

cin >> a;

h = (sqrt(3) / 2) \* a;

S = (h \* h) / sqrt(3);

r = a / (2 \* sqrt(3));

R = a / sqrt(3);

break;

case 2:

cin >> S;

h = sqrt(S \* sqrt(3));

a = h / (sqrt(3) / 2);

r = a / (2 \* sqrt(3));

R = a / sqrt(3);

break;

case 3:

cin >> h;

a = h / (sqrt(3) / 2);

S = (h \* h) / sqrt(3);

r = a / (2 \* sqrt(3));

R = a / sqrt(3);

break;

case 4:

cin >> r;

a = r \* (2 \* sqrt(3));

h = (sqrt(3) / 2) \* a;

S = (h \* h) / sqrt(3);

R = a / sqrt(3);

break;

case 5:

cin >> R;

a = R \* sqrt(3);

h = (sqrt(3) / 2) \* a;

S = (h \* h) / sqrt(3);

r = a / (2 \* sqrt(3));

break;

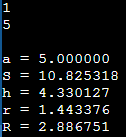
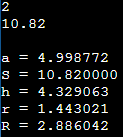
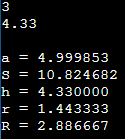
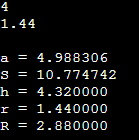
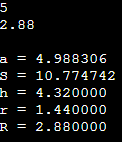
}

printf("\na = %lf \nS = %lf \nh = %lf \nr = %lf \nR = %lf", a, S, h, r, R);

return 0;

}

**Демонстрация результата работы программы**:

**Лабораторная работа №4**

**Циклы**

**Цель работы:** научиться строить алгоритмы и писать программы с циклами.

Вариант 22.

1) Определить, является ли заданное натуральное число совершенным, т.е. равным сумме все своих (положительных) делителей, кроме самого этого числа (например, число 6 совершенно: 6 =1+2+3).

**Текстовое описание решение задачи:**

Принять одно число(x) найти делители этого числа и сложить их.

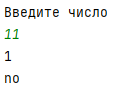
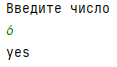
**Блок-схема**:



**Код на C++**:

#include **<iostream>**#include **<windows.h>  
  
using namespace** std;  
  
**int** main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
 **int** x, sum = 1;  
 cout << **"Введите число"** << endl;  
 cin >> x;  
 **for** (**int** i = 2; i < x / 2; i++){  
 **if** (x % i == 0)  
 sum += i;  
 }  
 cout << (sum == x ? **"yes"** : **"no"**) << endl;  
 system(**"pause"**);  
 **return** 0;  
}

**Тесты**:



**2) Дано натуральное n. Вычислить**:

**Текстовое описание решение задачи:**

**Считываем число n и с помощью цикла от 1 до n и перемножаю номер итерации на 2 и на произведение прошлых значений.**

**Блок схема:**



**Код** **на С++:**

#include **<iostream>**#include **<windows.h>**#include **<cmath>  
  
using namespace** std;  
  
**int** main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
 **int** n, y = 1;  
 cin >> n;  
 **for** (**int** i = 1; i <= n; i++){  
 y \*= i \* 2;  
 }  
 cout << y << endl;  
}

Тесты:



**3) Найти наименьший номер последовательности, для которого выполняется условие . Вывести на экран этот номер и все элементы**

**Блок-схема:**

****

**Код на С++:**

#include "stdafx.h"

#include <iostream>

#include <windows.h>

#include <cmath>

using namespace std;

int main() {

setlocale(LC\_ALL, "");

float e, x, a, new\_a;

float min = 3.40282e+038, min\_i;

cout << "Введите эпсилон и первый член последовательности" << endl;

cin >> e >> x;

if (x > 0.5 || x < 0){

cout << "x должен быть меньше 0.5 и больше 0" << endl;

system("pause");

main();

return 0;

}

a = x;

new\_a = a;

int i = 1;

do{

a = new\_a;

new\_a = (float) pow(a, 2) - 2 \* x;

++i;

cout << new\_a << endl;

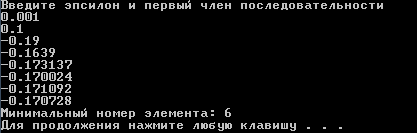
} while (abs(new\_a - a) >= e);

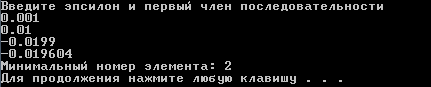
cout << "Минимальный номер элемента: " << --i << endl;

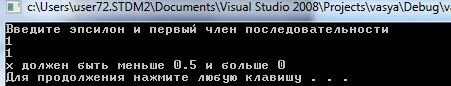
system("pause");

}

**Тесты:**







Лабораторная работа №5

«**Массивы**»

Цель работы: Научиться создавать алгоритмы с массивами.

Вариант 22:

1) Найти сумму и произведение первых пяти элементов массива

Текстовое описание решения: Создать массив с n элементами, заполнить его случайными числами и сложить все элементы которые на позиции меньше 5.

Блок схема:



Код на С++:

#include<iostream>

#include<cstdlib>

#include<ctime>

using namespace std;

int main(){

srand(time(0));

int k;

cout << "Введите количество элементов в массиве" << endl;

cin >> k;

int sum = 0, mult = 1;

int arr[k];

cout << "[";

for (int i = 0; i < k; i++){

arr[i] = rand() % 10;

if (i < 5){

sum += arr[i];

mult \*= arr[i];

}

if (i == k - 1)

cout << arr[i];

else

cout << arr[i] << ", ";

}

cout << "]" << endl;

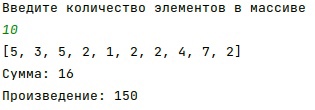
cout << "Сумма: " << sum << endl << "Произведение: " << mult << endl;

int i = 0;

return 0;

}

Тесты:



2) К элементам столбца k1 прибавить элементы столбца k2

Текстовое описание решения: Заполнить матрицу mхn случайными числами. Запросить у пользователя k1, k2 пройтись циклом по столбцу

Блок схема:



Код на С++:

#include<iostream>

#include<cstdlib>

#include<ctime>

using namespace std;

int main(){

srand(time(0));

int m, n;

cout << "Введите количство строк и столбцов в матрице" << endl;

cin >> m >> n;

if (m <= 0 || n <= 0){

cout << "Вы ввели плохие значения перезапустите программу" << endl;

return 0;

}

int matrix[m][n];

cout << "[" << endl;

for (int i = 0; i < m; i++){

cout << " [";

for (int j = 0; j < n; j++){

matrix[i][j] = rand() % 100;

cout << matrix[i][j] << (j < n - 1 ? ",\t": "");

}

cout << "]" << endl;

}

cout << "]" << endl;

cout << "Введите столбец к которому хотите прибавить(k1)" << endl;

int k1, k2;

cin >> k1;

cout << "Введите столбец который прибаляете(k2)" << endl;

cin >> k2;

for (int i = 0; i < n; i++){

matrix[i][k1 - 1] += matrix[i][k2 - 1];

}

cout << "[" << endl;

for (int i = 0; i < m; i++){

cout << " [";

for (int j = 0; j < n; j++){

cout << matrix[i][j] << (j < n - 1 ? ",\t": "");

}

cout << "]" << endl;

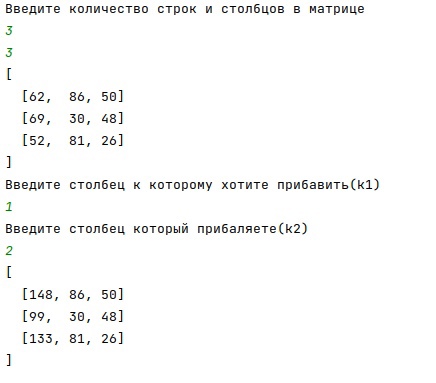
}

cout << "]" << endl;

return 0;

}

Тесты:



3) Вставить второй столбец после первого столбца, в котором все элементы положительны. Если такого столбца нет, то сообщить об этом.

Текстовое описание решения: Заполняем матрицы случайными значения(должна быть вероятность отрицательного числа). Проходим по столбцам, если элементы столбца положительны, то смещаем влево на один столбец от этого столбца все столбцы слева и заменяем левым столбец от положительного столбца элементами второго столбца.

Блок схема:



Код на С++:

#include<iostream>

#include<cstdlib>

#include<ctime>

using namespace std;

int main(){

srand(time(0));

int m, n;

cout << "Введите количество строк и столбцов в матрице" << endl;

cin >> m >> n;

if (m <= 0 || n <= 1){

cout << "Вы ввели плохие значения перезапустите программу" << endl;

return 0;

}

int matrix[m + 1][n + 1];

int copy\_matrix[n][m];

cout << "[" << endl;

for (int i = 0; i < m; i++){

cout << " [";

for (int j = 0; j < n; j++){

matrix[i][j] = rand() % 100 - 50;

cout << matrix[i][j] << (j < n - 1 ? ",\t": "");

copy\_matrix[i][j] = matrix[i][j];

}

cout << "]" << endl;

}

cout << "]" << endl;

for (int j = 0; j < n; j++){

bool is\_positive = true;

for (int i = 0; i < m; i++){

if (matrix[i][j] < 0){

is\_positive = false;

break;

}

}

if (is\_positive){

if (j != n - 1){

for (int jj = n; jj > j; jj--){

for (int i = 0; i < m; i++){

matrix[i][jj] = matrix[i][jj - 1];

}

}

}

for (int i = 0; i < m; i++){

matrix[i][j + 1] = copy\_matrix[i][1];

}

n++;

goto Section;

}

}

cout << "Столбец с положительными элементами не найден" << endl;

Section:

cout << "[" << endl;

for (int i = 0; i < m; i++){

cout << " [";

for (int j = 0; j < n; j++){

cout << matrix[i][j] << (j < n - 1 ? ",\t": "");

}

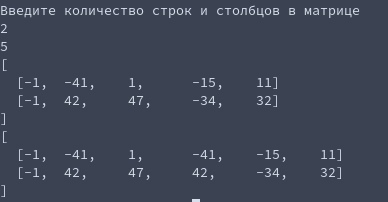
cout << "]" << endl;

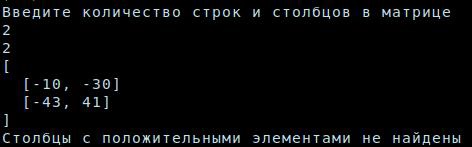
}

cout << "]" << endl;

}

Тесты:





Лабораторная работа №8

«**Структуры**»

22 Вариант

1. N учеников проходили тестирование по M тестам какого-либо предмета. Сколько очков набрал каждый ученик по всем темам?

Вычислить средний балл, полученный учениками, и разницу между лучшим результатом и средним баллом.

Текстовое описание задачи:

Код на С++:

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <vector>

#include <cmath>

using namespace std;

struct student {

int id = 0;

string name = " ";

string lastname = " ";

vector<int> marks;

string toString() {

string ms = "[";

for (auto el: marks) {

ms += to\_string(el) + (el != \*marks.rbegin() ? ", " : "]");

}

return to\_string(id) + " " + lastname + " " + name + " результаты теста:" + ms;

}

float average\_marks() {

float sum = 0;

for (auto el: marks) {

sum += el;

}

sum /= (float) marks.size();

return sum;

}

int get\_max\_mark() {

int max\_ = INT32\_MIN;

for (auto el: marks) {

if (el > max\_) max\_ = el;

}

return max\_;

}

};

int main() {

srand(time(0));

int N, M;

cout << "Введите количество учеников и количество тестов" << endl;

cin >> N >> M;

student students[N];

string random\_names[] = {"Вася", "Миша", "Марат", "Серёга", "Виктор", "Никита", "Вова", "Максим"};

string random\_lastnames[] = {"Петров", "Иванов", "Панков", "Гамуйло", "Хохланд", "Александров", "Сапогов",

"Мерлин"};

for (int i = 0; i < N; i++) {

student stud;

stud.id = i;

stud.lastname = random\_lastnames[rand() % 8];

stud.name = random\_names[rand() % 8];

stud.marks.resize(M);

for (int j = 0; j < M; j++) {

stud.marks[j] = rand() % 100;

}

cout << stud.toString() << endl;

students[i] = stud;

}

int absolute\_max = INT32\_MIN;

float all\_avarage = 0;

for (int i = 0; i < N; i++) {

int max\_ = students[i].get\_max\_mark();

float avarage = round(students[i].average\_marks() / 0.01) \* 0.01;

cout << students[i].lastname + " " + students[i].name + " Средний балл: " +

to\_string(avarage) +

" Лучший балл: " + to\_string(max\_) +

" Разница: " + to\_string(max\_ - avarage)

<< endl;

if (max\_ > absolute\_max)

absolute\_max = max\_;

all\_avarage += avarage;

}

all\_avarage /= N;

cout << "\n\nСамый большой балл: " << absolute\_max << endl << "Самый средний средний балл: " << all\_avarage << "\nРазница: " << absolute\_max - all\_avarage;

return 0;

}

Тесты:

1. Дан массив, содержащий информацию об учениках некоторой школы. Заполнить второй массив данными об учениках только девятых классов.

Код С++:

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

struct student{

int school\_class{};

char class\_name{};

string name;

string last\_name;

string toString(){

string new\_str = last\_name + " " + name + " " + to\_string(school\_class);

new\_str += class\_name;

return new\_str;

}

};

int main() {

srand(time(0));

int n, m;

string random\_names[] = {"Вася", "Миша", "Марат", "Серёга", "Виктор", "Никита", "Вова", "Максим"};

string random\_lastnames[] = {"Петров", "Иванов", "Панков", "Гамуйло", "Хохланд", "Александров", "Сапогов",

"Мерлин", "Бебров"};

string alph = "abcdefghijklmnopqrstuvwxyz";

cout << "Введите количество классов в параллели(до 26)" << endl;

cin >> n;

if (n > 26){

return 0;

}

cout << "Введите количество человек в классе?" << endl;

cin >> m;

student school[11 \* n \* m];

int index = 0;

cout << "Ученики из школы:" << endl;

for (int i = 0; i < 11; i++){

for (int g = 0; g < n; g++){

for (int p = 0; p < m; p++){

student new\_stud;

new\_stud.school\_class = i + 1;

new\_stud.class\_name = alph[g];

new\_stud.last\_name = random\_lastnames[rand() % 9];

new\_stud.name = random\_names[rand() % 8];

cout << new\_stud.toString() << endl;

school[index++] = new\_stud;

}

}

}

student new\_array[m];

index = 0;

cout << "\nУченики 9 класса:" << endl;

for (auto el: school){

if (el.school\_class == 9) {

new\_array[index++] = el;

cout << el.toString() << endl;

}

}

return 0;

}

**Лабораторная работа №15 "Простые классы"**

**Цель работы** - Реализовать класс без перегрузки операций, и без реализации конструкторов. Обязательно должны быть реализованы ввод с клавиатуры и вывод на экран. Операции реализуются в двух вариантах: как элементы класса и как дружественные функции.

**14 вариант**

**Условие задания** - Создать класс string для работы со строками, аналогичными строкам Turbo Pascal (строка представляется как массив 255 байтов, длина – в первом байте). Максимальный размер строки должен задаваться. Обязательно должны быть реализованы: длина строки, поиск подстроки в строке, удаление подстроки из строки, вставка подстроки в строку, сцепление двух строк.

**Код на С++**:

#include <iostream>

#include <cstring>

class string{

public:

explicit string(char\* src){

this->src = src;

this->size = strlen(src);

}

size\_t length() const{

return size;

}

char\* to\_string() const{

return src;

}

int findSubstr(char\* subString) {

size\_t sizesubStr = strlen(subString);

for (int i = 0; i < size; i++) {

if (subString[0] == src[i]) {

for (int j = 1; j < sizesubStr; j++) {

if (subString[j] != src[i + j]) {

goto nextiteration;

}

}

return i;

}

nextiteration:

continue;

}

return -1;

}

void deleteSubstr(char\* subString){

size\_t sizesubStr = strlen(subString);

int startIndex = this->findSubstr(subString);

if (startIndex == -1){

return;

}else{

char\* new\_str = new char[size - sizesubStr];

for (int i = 0; i < startIndex; i++) {

// std::cout << src[i];

new\_str[i] = src[i];

}

for (int i = startIndex; i < size - sizesubStr; i++) {

// std::cout << src[i];

new\_str[i] = src[i + sizesubStr];

}

src = new\_str;

size = strlen(src);

}

}

void insertSubStr(char\* subString, int startIndex = -1){

// char\* new\_str = src;

size\_t sizeSubString = strlen(subString);

// std::cout << (startIndex > size - 1) << std::endl;

if (startIndex > size - 1 && startIndex != -1)

throw std::overflow\_error("Index Error");

startIndex++;

char\* new\_str = new char[size + sizeSubString];

for (int i = 0; i < startIndex; i++) {

new\_str[i] = src[i];

}

for (int i = startIndex; i < startIndex + sizeSubString; i++){

new\_str[i] = subString[i - startIndex];

}

// std::cout << new\_str << std::endl;

for (size\_t i = startIndex + sizeSubString; i < size + sizeSubString; i++){

new\_str[i] = src[i - sizeSubString];

}

// std::cout << new\_str << std::endl;

src = new\_str;

size += sizeSubString;

}

string& operator+=(const string& str){

for (int i = 0; i < str.size; i++){

this->src[size + i] = str.src[i];

}

size += str.size;

src[size] = '\0';

return \*this;

}

friend string operator+(string str1, const string& str2){

str1 += str2;

return str1;

}

private:

char\* src;

size\_t size;

};

std::ostream &operator<<(std::ostream &os, const string &str){

return os << str.to\_string();

}

int main() {

char str[] = "Vasya Pankov";

string str2(str);

std::cout << str2.length() << std::endl;

str2.deleteSubstr((char\*) "Vasya");

std::cout << str2 << std::endl;

str2.insertSubStr((char\*)"Vasya", -1);

std::cout << str2 << std::endl;

str2 += string((char \*)" is the best man");

std::cout << str2 << std::endl;

std::cout << str2.length() << std::endl;

std::cout << (str2 + string((char \*)". "));

}

Результат выполнения:

12

Pankov

Vasya Pankov

Vasya Pankov is the best man

28

Vasya Pankov is the best man.

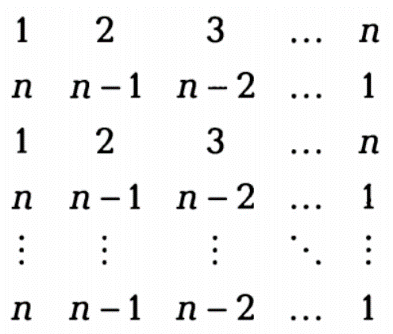
**Пояснения**:

Создан класс string, который немного не соответствует требованиям, но реализован лучше. У класса есть приватные свойства src(сам текст) и size(его размер). При инициализации мы задаём просто строчку, для поиска опр. подстроки мы просто в параметре метода вводим саму подстроку. Для добавления подстроки нужна указать индекс символа, который будет слева от начала, если надо вставит начало, то -1. Для удаления подстроки просто введите её(удалится только первое вхождение). Для объединения двух строчек их можно просто сложить. Также реализован обычный вывод через cout.

**Лабораторная работа №16 "Массивы"**

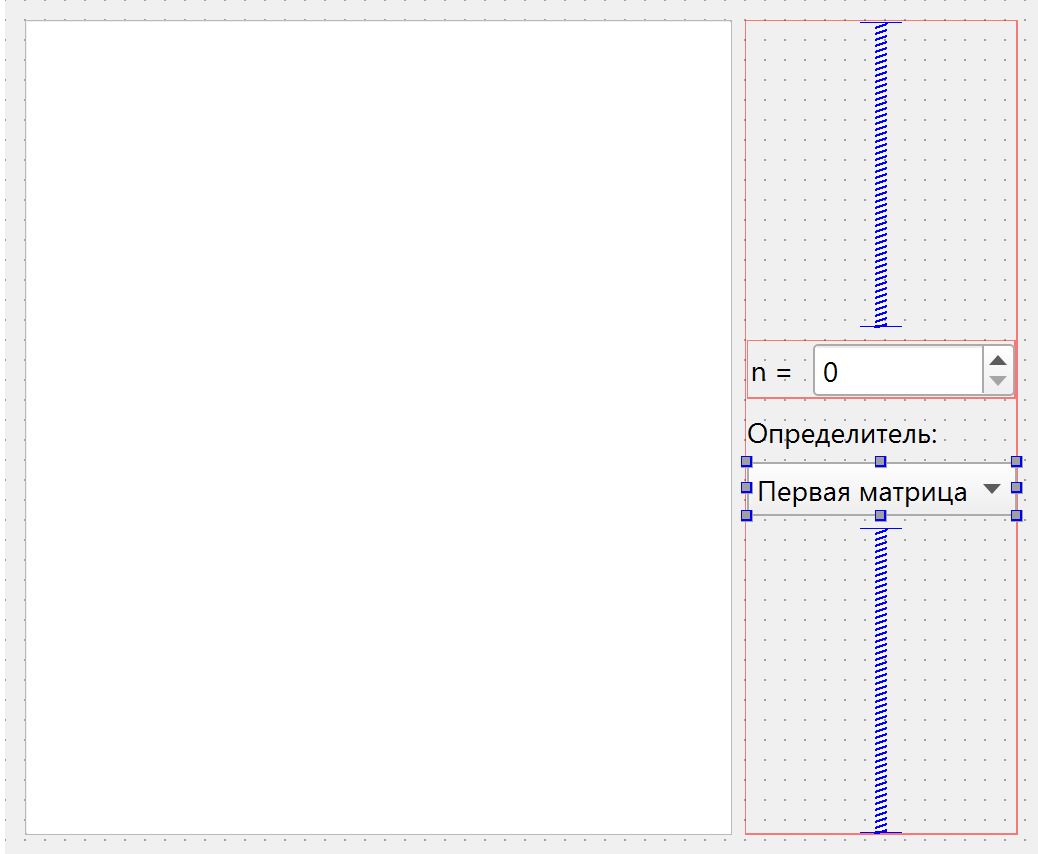
Вариант 1.

1. Сформировать квадратную матрицу по образцу (заполнить и вывести на экран):





Дизайн приложения:



Код приложения:

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

ui->label\_2->*setVisible*(false);

table = ui->tableWidget;

QSpinBox::connect(ui->spinBox, &QSpinBox::valueChanged, this, &MainWindow::OnNChanged);

QComboBox::connect(ui->comboBox, &QComboBox::currentIndexChanged, this, &MainWindow::OnMatrixTypeChanged);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::**OnNChanged**(int n){

this->n = n;

//matrix.clear();

if (type == 0){

table->setRowCount(n);

table->setColumnCount(n);

for (int i = 0; i < n; i++){

table->setColumnWidth(i, 5);

for (int j = 0; j < n; j++){

QTableWidgetItem\* item = new QTableWidgetItem();

item->setTextAlignment(Qt::AlignCenter);

if (i % 2 == 0){

item->setText(QString::number(j + 1));

matrix[i][j] = j + 1;

} else{

item->setText(QString::number(n - j));

matrix[i][j] = n - j;

}

table->setItem(i, j, *item*);

}

}

} else if (type == 1){

table->setRowCount(2 \* n);

table->setColumnCount(2 \* n);

for (int i = 0; i < n; i++){

table->setColumnWidth(i, 5);

for (int j = 0; j < n; j++){

QTableWidgetItem\* item = new QTableWidgetItem();

item->setTextAlignment(Qt::AlignCenter);

item->setText(QString::number(1));

table->setItem(i, j, *item*);

}

}

for (int i = 0; i < n; i++){

table->setColumnWidth(i, 5);

for (int j = n; j < 2 \* n; j++){

QTableWidgetItem\* item = new QTableWidgetItem();

item->setTextAlignment(Qt::AlignCenter);

item->setText(QString::number(2));

table->setItem(i, j, *item*);

}

}

for (int i = n; i < 2 \* n; i++){

table->setColumnWidth(i, 5);

for (int j = n; j < 2 \* n; j++){

QTableWidgetItem\* item = new QTableWidgetItem();

item->setTextAlignment(Qt::AlignCenter);

item->setText(QString::number(4));

table->setItem(i, j, *item*);

}

}

for (int i = n; i < 2 \* n; i++){

table->setColumnWidth(i, 5);

for (int j = 0; j < n; j++){

QTableWidgetItem\* item = new QTableWidgetItem();

item->setTextAlignment(Qt::AlignCenter);

item->setText(QString::number(3));

table->setItem(i, j, *item*);

}

}

}

}

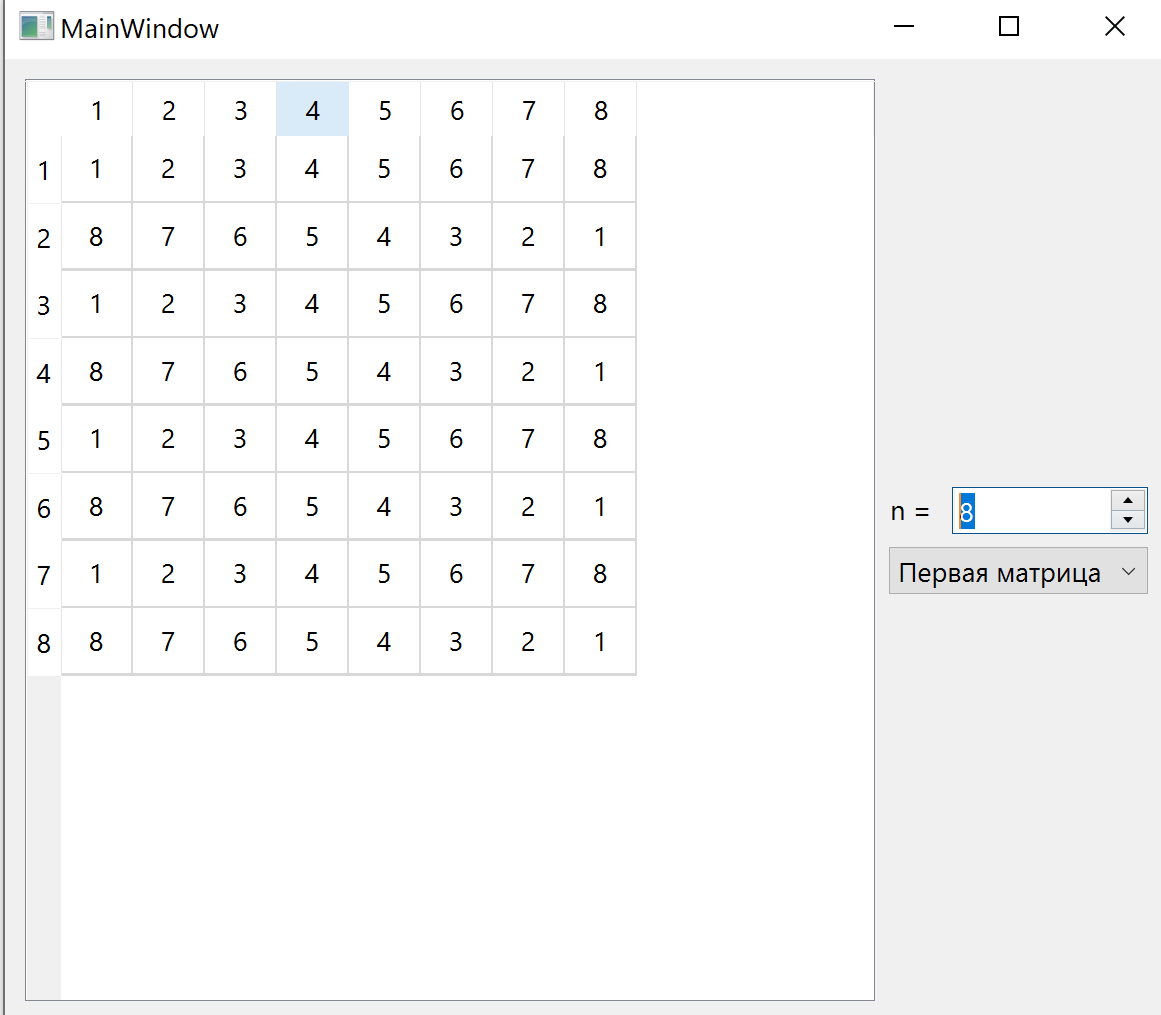
void MainWindow::**OnMatrixTypeChanged**(int n){

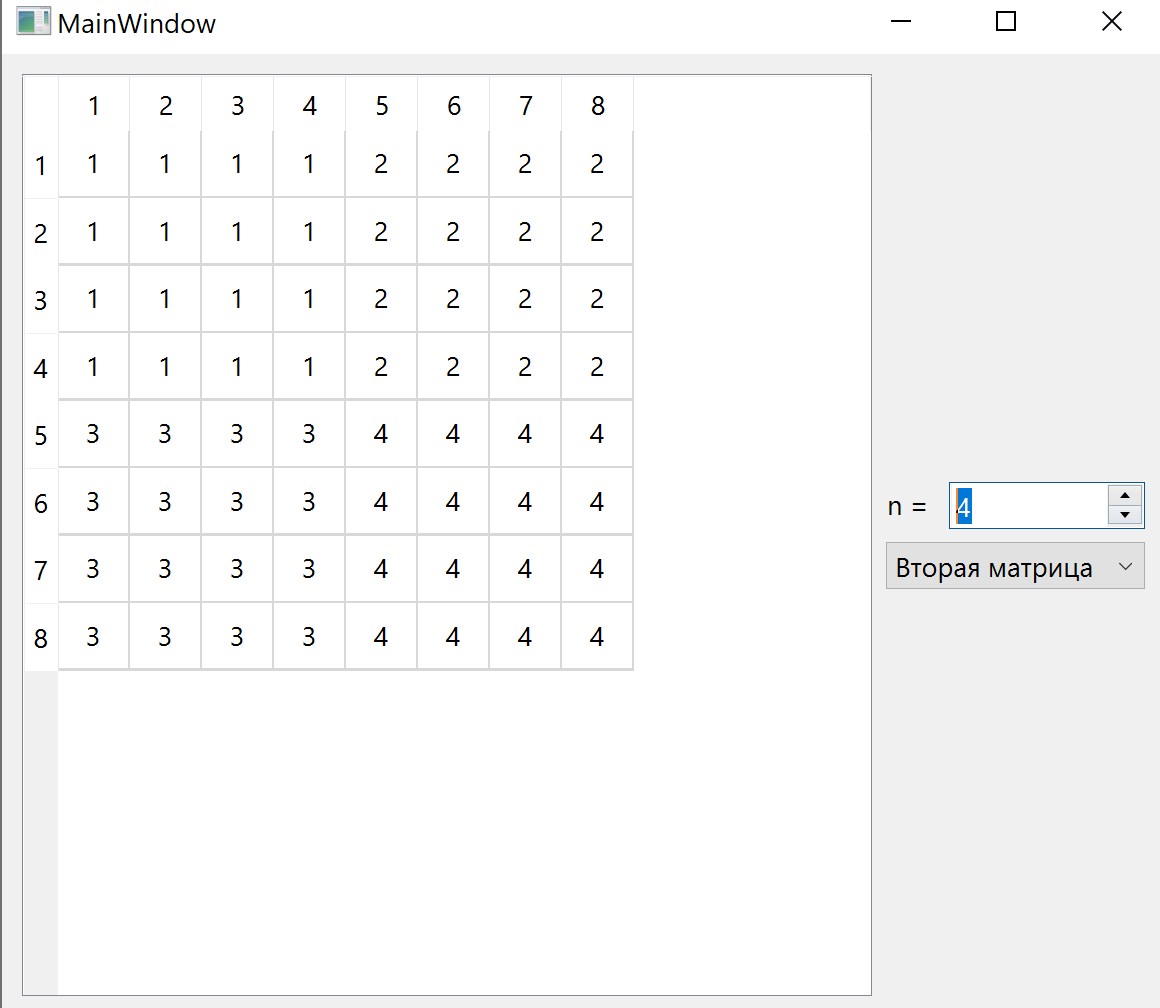
this->type = n;

MainWindow::OnNChanged(this->n);

}

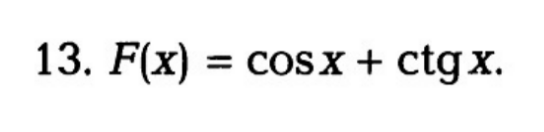
Демонстрация работа:



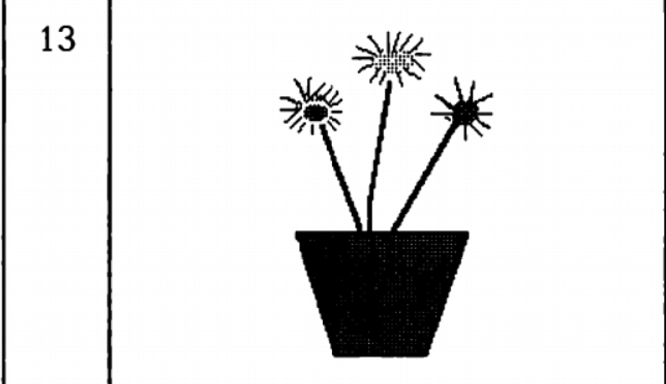


**Лабораторная работа № 17**

1.1. Задание 1 Построить график заданной функции.

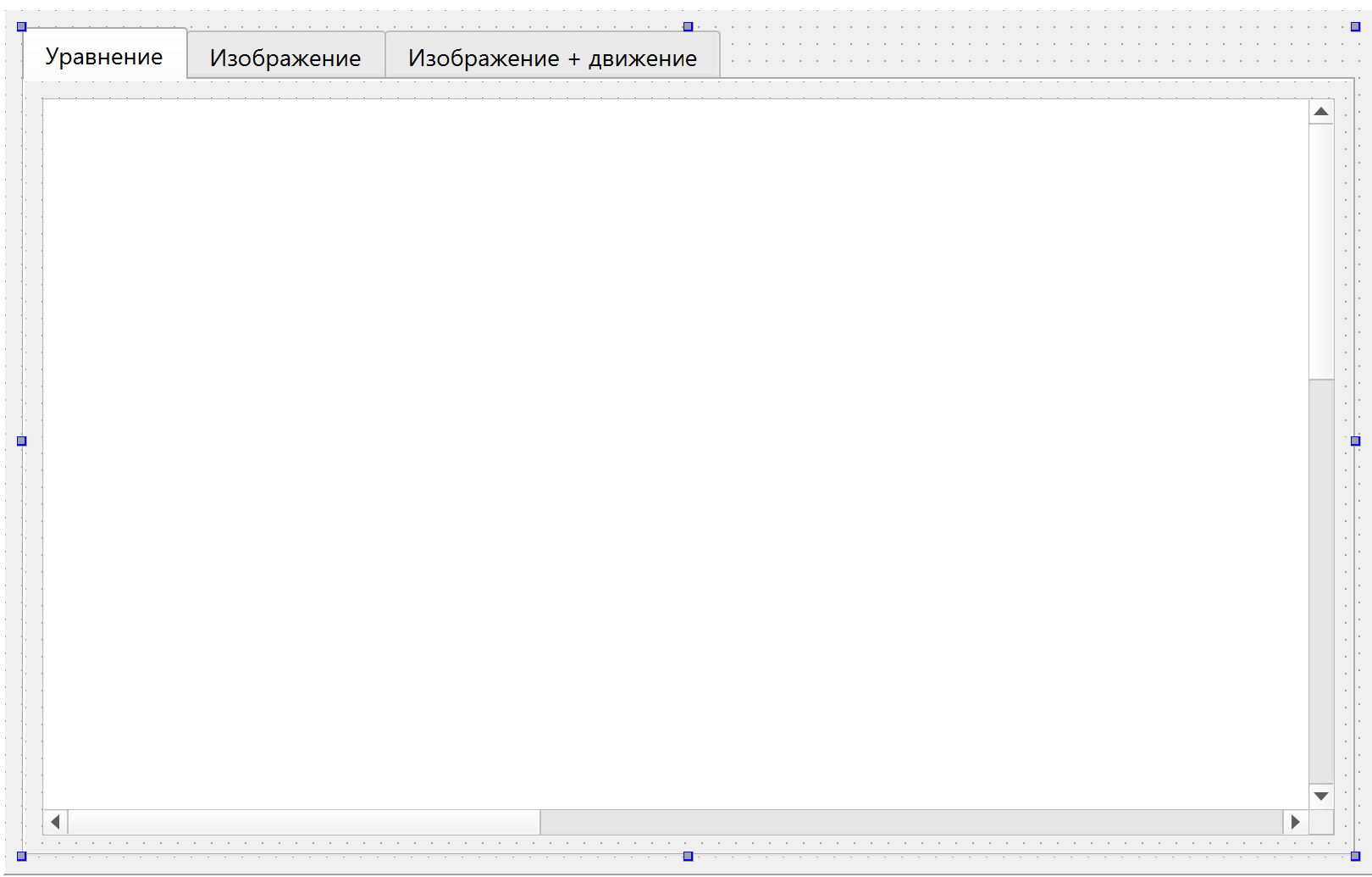


1.2. Задание 2 Нарисовать изображения средствами выбранного фреймворка(для меня QT)



Выполнение заданий

Интерфейс приложения:



Код отображения графика:

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

#include "QtMath"

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

// scene(this);

// scene.addSceneRect(QRectF(0, 0, 500, 500)); // 760x440

scene = new QGraphicsScene(this);

QBrush qbrush(QColor::fromRgbF(0, 0, 0, 0));

QPen pen(QColor::fromRgbF(1, 0, 0, 1));

pen.setWidth(2);

// scene->addRect(QRect(50, 50, 100, 100));

scene->addLine(0, 540, 1920, 540);

scene->addLine(960, 0, 960, 1080);

QPointF last\_point(0, 220);

QPointF point;

for (double i = 0; i < 1920; i += 1.0){

point = QPointF(i, 540 - cos(i / 100) + cos(i / 100) / sin(i / 100));

scene->addLine(QLineF(last\_point, point), pen);

last\_point = point;

}

ui->graphicsView->setScene(scene);

second\_scene = new QGraphicsScene(this);

polygon = new QPolygonF();

\*polygon << QPointF(299, 251) << QPointF(322, 347) << QPointF(407, 347) << QPointF(430, 251) << QPointF(299, 251);

second\_scene->addPolygon(\*polygon, QPen(QColor::fromRgb(248, 207, 255)), QBrush(QColor::fromRgb(248, 207, 255)));

second\_scene->addLine(QLine(307, 106, 356, 251));

second\_scene->addLine(QLine(454, 110, 384, 251));

second\_scene->addLine(QLine(375, 100, 365, 250));

second\_scene->addEllipse(QRectF(307 - 10, 106 - 10, 20, 20), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

second\_scene->addEllipse(QRectF(454 - 10, 110 - 10, 20, 20), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

second\_scene->addEllipse(QRectF(375 - 10, 100 - 10, 20, 20), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

ui->graphicsView\_2->setScene(second\_scene);

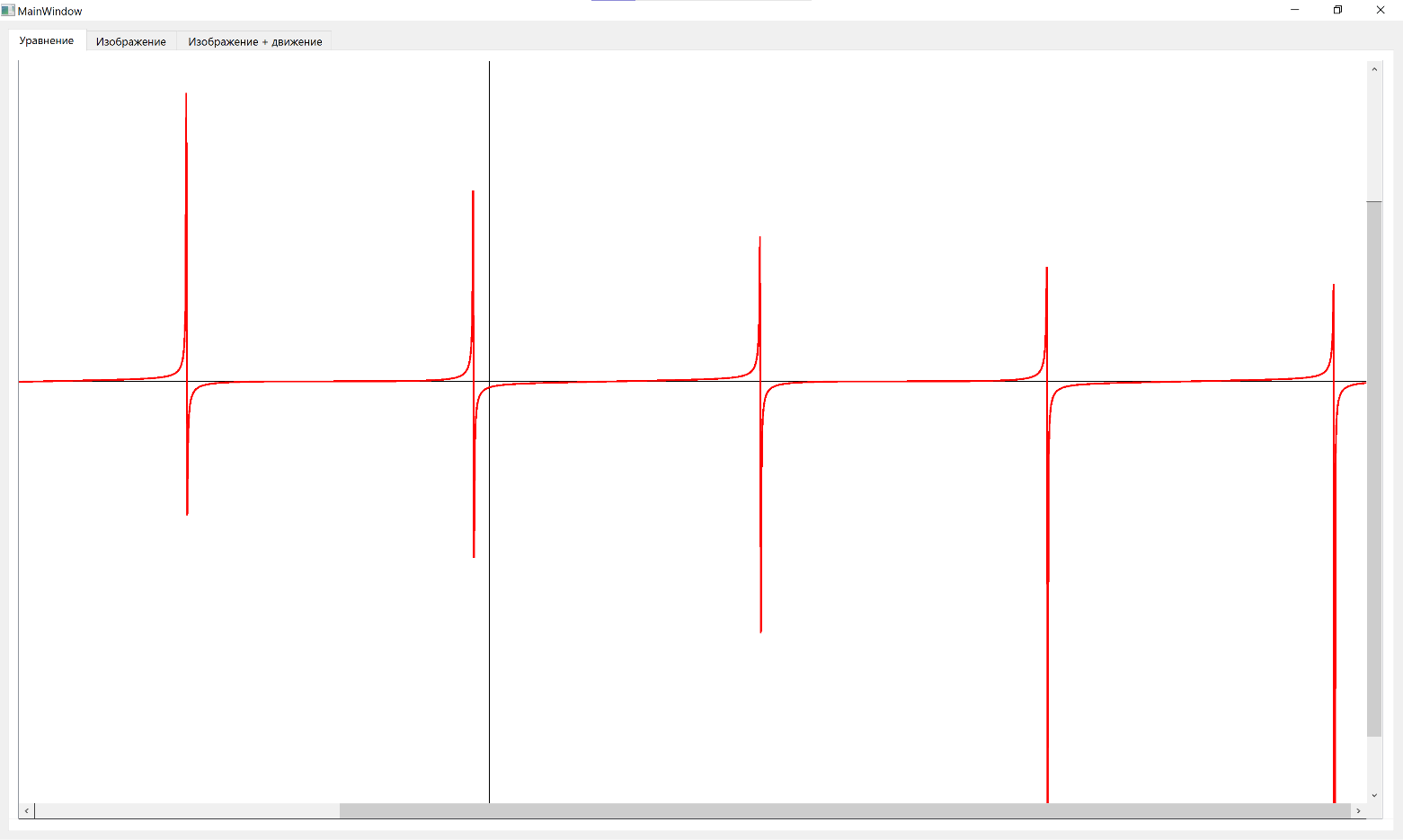
}

MainWindow::~MainWindow()

{

delete ui;

}

Отображение работы приложения: 

**Лабораторная работа № 18**

**Задание.** Для изображения (или его элемента) из прошлой работы написать функцию. В качестве параметров в функцию можно передать указатель на графический объект (или элемент управления, где будет построено изображение, например: PictureBox, Form), координаты одной ключевой точки, размер изображения, перо, кисть и другие необходимые параметры.

**Лабораторная работа № 19**

**Задание**. Доработать приложение, которое «умеет» рисовать и скрывать графическое изображение так, чтобы с помощью клавиш управления курсором можно было перемещать это изображение.

**Лабораторная работа № 21.**

Создание анимации. Элемент управления Timer.

**Задание**. Доработать приложение, которое «умеет» перемещать картинку с помощью клавиш управления курсором, так, чтобы после нажатия на клавишу Enter запускался Timer и картинка начинала движение, а кнопка Esc – останавливала его.

Итоговый код по этим трём работам:

Mainwindow.h:

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QGraphicsScene>

#include <mygraphicsscene.h>

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **MainWindow**; }

QT\_END\_NAMESPACE

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

**MainWindow**(QWidget \*parent = nullptr);

~***MainWindow***();

MyGraphicsScene \*scene;

QPolygonF \*polygon;

QGraphicsScene \*second\_scene;

QGraphicsScene \*third\_scene;

QTimer \*timer;

QPair<double, double> \*drw\_now\_pos;

void **redraw**();

void **moveImg**();

QAction \*Rigth, \*Left, \*Forward, \*Backward;

protected:

void ***keyPressEvent***(QKeyEvent\* event);

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

mygraphicsscene.h:

#ifndef MYGRAPHICSSCENE\_H

#define MYGRAPHICSSCENE\_H

#include <QGraphicsScene>

class **MyGraphicsScene** : public QGraphicsScene

{

public:

explicit **MyGraphicsScene**(QObject \*parent = nullptr);

private:

void ***mousePressEvent***(QGraphicsSceneMouseEvent \*event) override;

};

#endif // MYGRAPHICSSCENE\_H

mainwindow.cpp:

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

#include "QtMath"

#include <QKeyEvent>

#include <QTimer>

#include <mygraphicsscene.h>

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

// таймер здесь

timer = new QTimer(this);

connect(timer, &QTimer::timeout, this, &MainWindow::moveImg);

scene = new MyGraphicsScene(this);

QBrush qbrush(QColor::fromRgbF(0, 0, 0, 0));

QPen pen(QColor::fromRgbF(1, 0, 0, 1));

pen.setWidth(2);

scene->addLine(0, 540, 1920, 540);

scene->addLine(960, 0, 960, 1080);

QPointF last\_point(0, 220);

QPointF point;

for (double i = 0; i < 1920; i += 1.0){

point = QPointF(i, 540 - cos(i / 100) + cos(i / 100) / sin(i / 100));

scene->addLine(QLineF(last\_point, point), pen);

last\_point = point;

}

ui->graphicsView->setScene(*scene*);

// Flowers without moves

second\_scene = new QGraphicsScene(this);

polygon = new QPolygonF();

\*polygon << QPointF(299, 251) << QPointF(322, 347) << QPointF(407, 347) << QPointF(430, 251) << QPointF(299, 251);

second\_scene->addPolygon(\*polygon, QPen(QColor::fromRgb(248, 207, 255)), QBrush(QColor::fromRgb(248, 207, 255)));

second\_scene->addLine(QLine(307, 106, 356, 251));

second\_scene->addLine(QLine(454, 110, 384, 251));

second\_scene->addLine(QLine(375, 100, 365, 250));

second\_scene->addEllipse(QRectF(307 - 10, 106 - 10, 20, 20), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

second\_scene->addEllipse(QRectF(454 - 10, 110 - 10, 20, 20), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

second\_scene->addEllipse(QRectF(375 - 10, 100 - 10, 20, 20), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

ui->graphicsView\_2->setScene(*second\_scene*);

// Flowers with moves

drw\_now\_pos = new QPair<double, double>(0, 0);

third\_scene = new QGraphicsScene(this);

redraw();

//ui->graphicsView\_3->grabKeyboard();

}

void MainWindow::***keyPressEvent***(QKeyEvent\* event){

switch ( event->key() ) {

case Qt::Key\_W:

if (drw\_now\_pos->second > 0){

drw\_now\_pos->second -= 5;

redraw();

}

break;

case Qt::Key\_S:

if (drw\_now\_pos->second + 159 < 1000){

drw\_now\_pos->second +=5;

redraw();

}

break;

case Qt::Key\_D:

if (drw\_now\_pos->first + 130 < 1000){

drw\_now\_pos->first += 5;

redraw();

}

break;

case Qt::Key\_A:

if (drw\_now\_pos->first > 0){

drw\_now\_pos->first -= 5;

redraw();

}

break;

case Qt::Key\_E:

timer->start(100);

break;

case Qt::Key\_Escape:

timer->stop();

break;

default:

event->ignore();

break;

}

}

void MainWindow::**moveImg**(){

drw\_now\_pos->first += 4;

drw\_now\_pos->second += 4;

redraw();

}

void MainWindow::**redraw**(){

third\_scene->clear();

polygon = new QPolygonF();

\*polygon << QPointF(drw\_now\_pos->first, drw\_now\_pos->second + 159) << QPointF(drw\_now\_pos->first + 23, drw\_now\_pos->second + 255) << QPointF(drw\_now\_pos->first + 108, drw\_now\_pos->second + 255) << QPointF(drw\_now\_pos->first + 131, drw\_now\_pos->second + 158) << QPointF(drw\_now\_pos->first, drw\_now\_pos->second + 159);

third\_scene->addPolygon(\*polygon, QPen(QColor::fromRgb(248, 207, 255)), QBrush(QColor::fromRgb(248, 207, 255)));

third\_scene->addLine(QLine(drw\_now\_pos->first + 9, drw\_now\_pos->second + 14, drw\_now\_pos->first + 58, drw\_now\_pos->second + 159));

third\_scene->addLine(QLine(drw\_now\_pos->first + 77, drw\_now\_pos->second + 8, drw\_now\_pos->first + 67, drw\_now\_pos->second + 159));

third\_scene->addLine(QLine(drw\_now\_pos->first + 156, drw\_now\_pos->second + 18, drw\_now\_pos->first + 86, drw\_now\_pos->second + 159));

third\_scene->addEllipse(QRectF(drw\_now\_pos->first + 1, drw\_now\_pos->second + 6, 16, 16), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

third\_scene->addEllipse(QRectF(drw\_now\_pos->first + 69, drw\_now\_pos->second, 16, 16), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

third\_scene->addEllipse(QRectF(drw\_now\_pos->first + 148, drw\_now\_pos->second + 10, 16, 16), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

ui->graphicsView\_3->setScene(*third\_scene*);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

mygraphicscene,cpp:

#include "mygraphicsscene.h"

#include <QGraphicsSceneMouseEvent>

#include <qdebug.h>

MyGraphicsScene::**MyGraphicsScene**(QObject \*parent)

: QGraphicsScene{*parent*}

{

}

void MyGraphicsScene::***mousePressEvent***(QGraphicsSceneMouseEvent \* e)

{

QPair<double, double> \*drw\_now\_pos = new QPair<double, double>(e->scenePos().x(), e->scenePos().y());

QPolygonF\* polygon = new QPolygonF();

\*polygon << QPointF(drw\_now\_pos->first, drw\_now\_pos->second + 159) << QPointF(drw\_now\_pos->first + 23, drw\_now\_pos->second + 255) << QPointF(drw\_now\_pos->first + 108, drw\_now\_pos->second + 255) << QPointF(drw\_now\_pos->first + 131, drw\_now\_pos->second + 158) << QPointF(drw\_now\_pos->first, drw\_now\_pos->second + 159);

this->addPolygon(\*polygon, QPen(QColor::fromRgb(248, 207, 255)), QBrush(QColor::fromRgb(248, 207, 255)));

this->addLine(QLine(drw\_now\_pos->first + 9, drw\_now\_pos->second + 14, drw\_now\_pos->first + 58, drw\_now\_pos->second + 159));

this->addLine(QLine(drw\_now\_pos->first + 77, drw\_now\_pos->second + 8, drw\_now\_pos->first + 67, drw\_now\_pos->second + 159));

this->addLine(QLine(drw\_now\_pos->first + 156, drw\_now\_pos->second + 18, drw\_now\_pos->first + 86, drw\_now\_pos->second + 159));

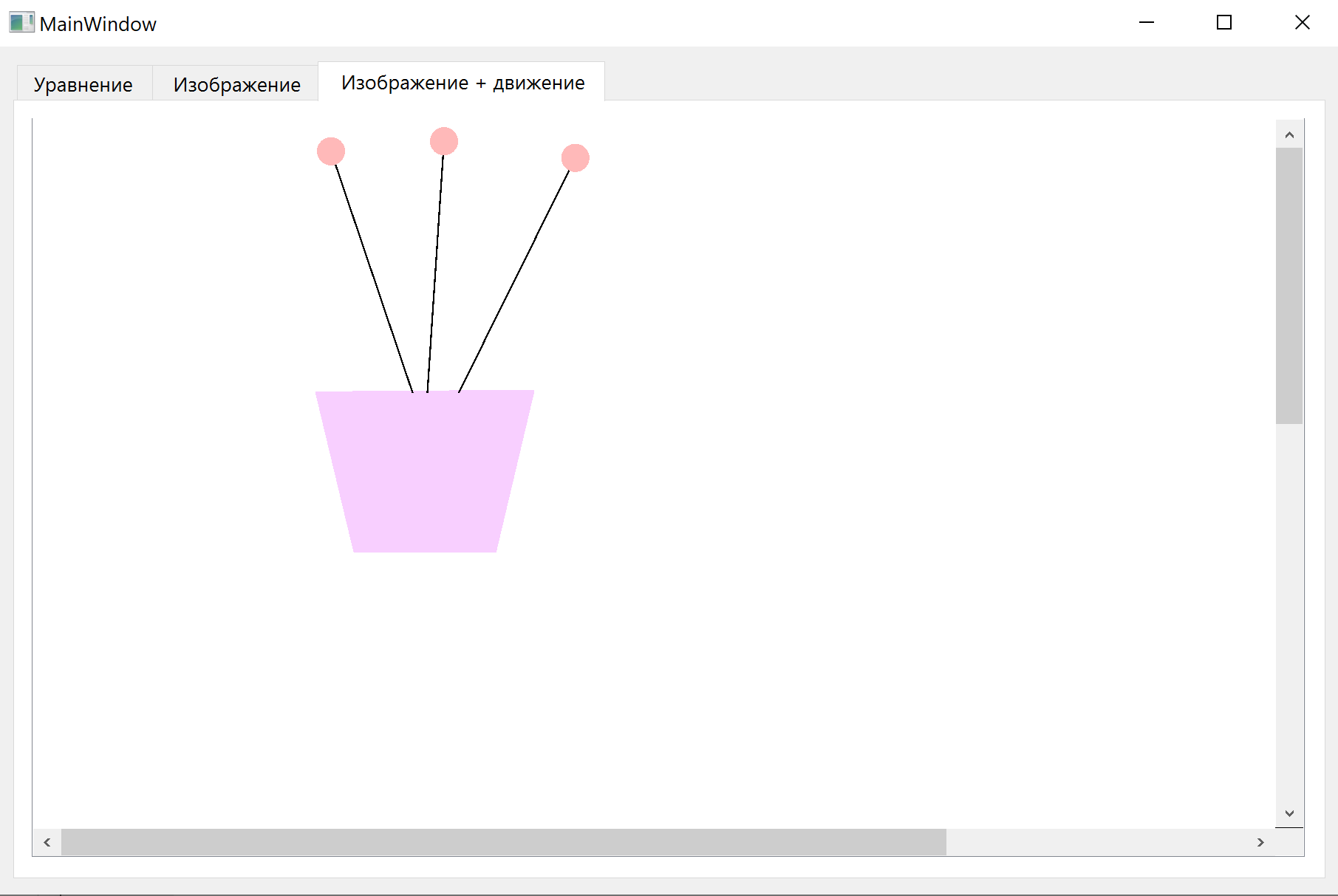
this->addEllipse(QRectF(drw\_now\_pos->first + 1, drw\_now\_pos->second + 6, 16, 16), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

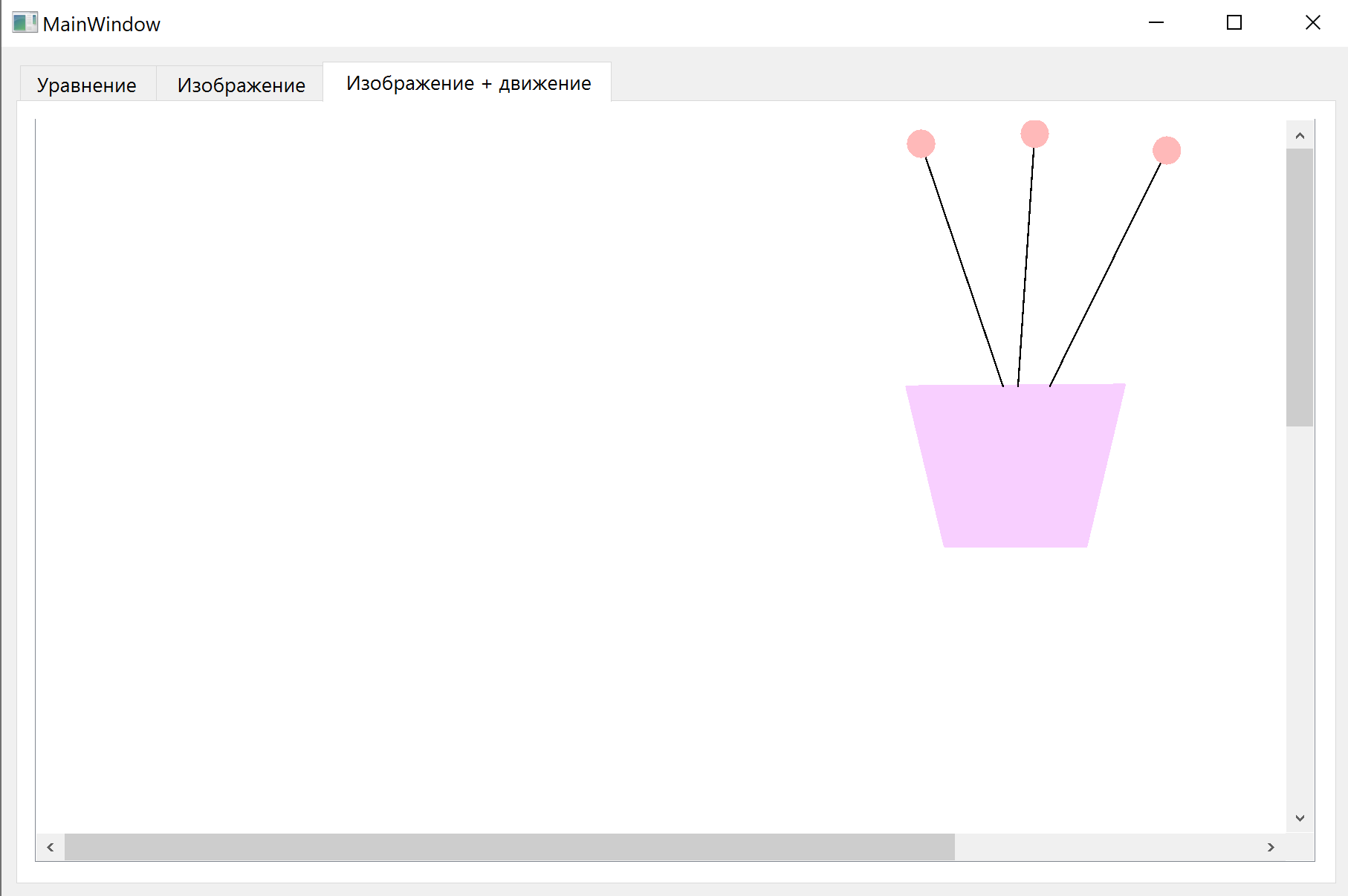
this->addEllipse(QRectF(drw\_now\_pos->first + 69, drw\_now\_pos->second, 16, 16), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

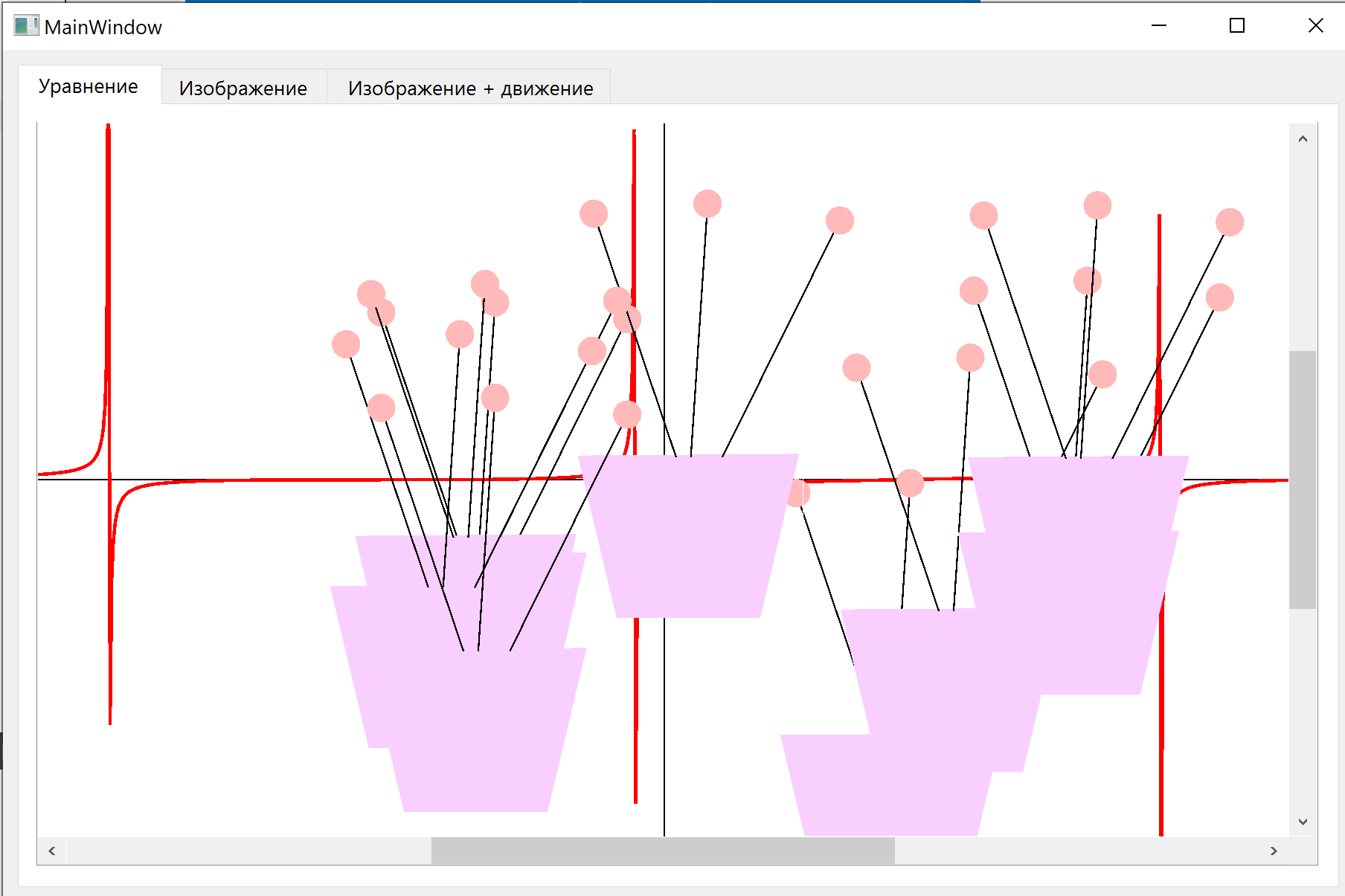
this->addEllipse(QRectF(drw\_now\_pos->first + 148, drw\_now\_pos->second + 10, 16, 16), QPen(QColor::fromRgb(255, 185, 185)), QBrush(QColor::fromRgb(255, 185, 185)));

}

Демонстрация работы приложения:







**Лабораторная работа № 25**

**Работа с файлами в Visual Studio C++. Использование файлов для хранения данных.**

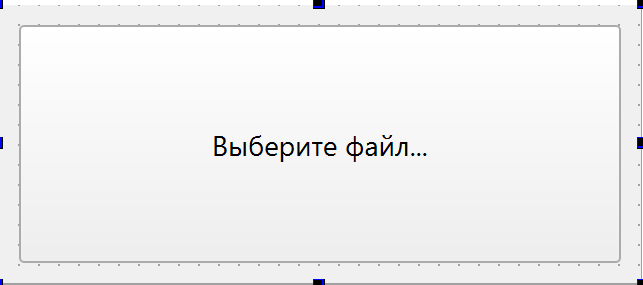
**Цель работы**: изучить приемы работы с файлами и способы создания файлов. Получить практические навыки по использованию файлов для хранения информации.

**Индивидуальное задание**:

**Вариант №13**

Дан файл, содержащий текст на русском языке и некоторая буква. Найти слово, содержащее наибольшее количество указанных букв.

Интерфейс приложения:



Код приложения:

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

#include <QPushButton>

#include <QFileDialog>

#include <QMessageBox>

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

QPushButton::connect(ui->pushButton, &QPushButton::pressed, this, &MainWindow::choosefile);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::**choosefile**(){

QString fileName = QFileDialog::getOpenFileName(this,

tr("Open Text file"), "", tr("Text Files (\*.txt \*.md \*.org)"));

QMessageBox msg\_box;

if (fileName.isNull() || fileName.isEmpty())

return;

else{

QString str = "Исходный текст:<br> ";

QFile file(fileName);

if (!file.*open*(QIODevice::ReadOnly | QIODevice::Text))

return;

QTextStream in(*&file*);

QString line;

QString text;

while (!in.atEnd()) {

line = in.readLine();

text += line + "\n ";

}

QChar chr = text[text.length() - 3];

str += text.sliced(0, text.length() - 3);

QString max\_word = "";

int max\_how = 0;

for (auto word: text.split(' ')){

if (max\_how < word.count(chr, Qt::CaseInsensitive)){

max\_how = word.count(chr, Qt::CaseInsensitive);

max\_word = word;

}

}

str += tr("Символ: ") + chr + tr("<br>");

str += tr("Слово с наиюольшим повторением данного символа: ") + max\_word;

str.replace("\n", "<br>");

msg\_box.setText(str);

}

msg\_box.*exec*();

}

Демонстрация работы:

